Instant Selenium Node.js Tests

If you want to get up and running with Sauce Labs but don't have any Selenium test scripts, you can use the ones in this topic to see how how it works. All you need to do is add your Sauce Labs username and password, and then run the script from the local machine where you have your Selenium environment set up. When the script runs, it will connect to Sauce Labs using your authentication credentials, launch the browser set in the test script, load the Sauce Labs demo website, and then close the browser and quit the test.

After running the test, you can log into the Sauce Labs web interface to see the test listed on your dashboard, and view information about the test, including video, by clicking on the test name to view the Test Details page.

If you want to experiment with the sample script further, try using the Platform Configurator to change the Desired Capabilities of the test, or substitute the URL of the web app you want to test for the `sauce.demo` URL.

Example Only

The code in this topic is presented as an example only, since your tests and testing environments may require specialized scripting. This information should be taken only as an illustration of how you could set up your tests with Sauce Labs, and is not directly supported by Sauce.

Prerequisites

- You should have a Sauce Labs account
- You should have a node.js Selenium environment already set up on the local machine where you'll launch your tests. If you need help with this, check out the ReadMe in the JavaScript sample script repository on GitHub.
- You should review the Best Practices for Running Tests

Instant Test

Download this script from the GitHub repo to your local machine, provide the information for `SAUCE_USERNAME` and `SAUCE_ACCESS_CODE` as indicated in the script, and then save and run the script. If you log in to saucelabs.com before you run the script, you'll be able to watch the script's progress as it executes.

Click here to view the example script

Clone from: https://github.com/saucelabs-training/demo-js/blob/master/web-tests/instant-sauce-mocha-test1.js

web-tests/instant-sauce-mocha-test1.js (source from Sauce Labs Training JavaScript)

```
var webdriver = require('selenium-webdriver'),
    /* Change the username and accessKey to your Saucelabs.com
credentials */
    username = "SAUCE_USERNAME",
    accessKey = "SAUCE_ACCESS_KEY",
    /* Base URL sets the target test application */
    baseUrl = "https://www.saucedemo.com",
    /* driver instantiates via callback */
    driver;

    /* Describe is a way to group your tests together and set test
suite parameters like timetous */
describe('Instant Sauce Test Module 1', function() {
    this.timeout(40000);
    /* it represents an actual test, the parameters are the title of
the test case */
    it('should-open-safari', function (done) {
        /* Instantiate a WebDriver and set browser capabilities */
        driver = new webdriver.Builder().withCapabilities({
            'browserName': 'safari',
            'platform': 'macOS 10.13',
            'version': '11.1',
            /* Pass Sauce User Name and Access Key */
            'username': username,
            'accessKey': accessKey,
            'build': 'Onboarding Sample App - NodeJS',
            'name': '1-first-test',
        }).usingServer("http://" + username + ":" + accessKey +
            "@ondemand.saucelabs.com:80/wd/hub").build();
        /* The driver navigates to the target application, stored in
this variable baseUrl*/
        driver.get(baseUrl);
        /* The driver grabs the title of the web page and displays it
in your console */
        driver.getTitle().then(function (title) {
            console.log("title is: " + title);
        });
        /* This tears down the current WebDriver session and ends the
test method*/
        driver.quit();
        done();
    });
});
```

To find your Sauce Labs access key:

1. Sign in to https://saucelabs.com with the username you set up when you created your account. You will use this same username in your test script.
2. To find your access key:
    1. Click your name in the **Account Profile** menu in the upper-right corner.
    2. Click **User Settings**.
    3. Scroll down to **Access Key** and click **Show**.

4. Enter the password you use to sign in to Sauce Labs to view the access key.
5. Click the **Copy** icon.

IP Ranges and Data Center Endpoints
Sauce Labs has data centers in both the US and EU. You can access Sauce Labs services from either location by providing the appropriate URL endpoints, and whitelisting their associated IP ranges.

- **Status Page for the US Data Center**: https://status.saucelabs.com
- **Status Page for the EU Data Center**: http://status.eu-central-1.saucelabs.com/

Click here to view US Data Center Endpoints

Virtual Device Cloud: `https://ondemand.saucelabs.com/wd/hub`

Sauce Connect Proxy: `https://saucelabs.com/rest/v1` (do not need to specify, Sauce Connect Proxy will use this by default)

SSO:

- `https://saucelabs.com/sso/metadata` for **Signing/Encryption Certificate**
- `https://saucelabs.com/sso/acs` for **Entity AssertionConsumeURL**

REST API:

- Virtual Devices and Desktops: `api.us-west-1.saucelabs.com/v1`
- Real Devices: `api.us-west-1.saucelabs.com/v1/rdc`

IP Ranges:

`162.222.72.0/21` (this is equivalent to the range `162.222.72.1 – 162.222.79.254`)
`66.85.48.0/21` (this is equivalent to the range `66.85.48.0 - 66.85.55.255`)

Click here to view EU Data Center endpoints

For more information, including framework configuration for the European data center, check out Sauce Labs European Data Center Configuration Information.

Virtual Device Cloud: `https://ondemand.eu-central-1.saucelabs.com/wd/hub`

Sauce Connect Proxy: `https://eu-central-1.saucelabs.com/rest/v1` (specify with **-x** argument)

SSO:

- `https://eu-central-1.saucelabs.com/sso/metadata` for **Signing/Encryption Certificate**
- `https://eu-central-1.saucelabs.com/sso/acs` for **Entity AssertionConsumeURL**

REST API: `https://eu-central-1.saucelabs.com/rest`

IP Range:

`185.94.24.0/22` (this is equivalent to the range `185.94.24.0 - 185.94.27.255`

Instant Test with Environment Variables

This example script follows the best practice of using environment variables in place of hardcoded authentication credentials. Once you've been able to successfully run the instant test with hardcoded credentials, you can use this test to make sure that you've correctly set up your environment variables for authentication.

Click here to view the example script

Clone from: https://github.com/saucelabs-training/demo-js/blob/master/web-tests/instant-sauce-mocha-test2.js

web-tests/instant-sauce-mocha-test2.js (source from Sauce Labs Training JavaScript)

```
var webdriver = require('selenium-webdriver'),
    /* Use a run configuration and/or a bash profile to set your
environment variables,
    for more information on how to do this, please visit:
    https://wiki.saucelabs.com/display/DOCS/Best+Practice%
3A+Use+Environment+Variables+for+Authentication+Credentials
     */
    username = process.env.SAUCE_USERNAME,
    accessKey = process.env.SAUCE_ACCESS_KEY,

    /* Change the baseURL to your application URL */
    baseUrl = "https://www.saucedemo.com",
    driver;

describe('Instant Sauce Test Module 1', function() {
    this.timeout(40000);
    it('should-open-safari', function (done) {
        driver = new webdriver.Builder().withCapabilities({
            'browserName': 'safari',
            'platform': 'macOS 10.13',
            'version': '11.1',
            'username': username,
            'accessKey': accessKey,
            'build': 'Onboarding Sample App - NodeJS',
            'name': '2-user-site',
        }).usingServer("http://" + username + ":" + accessKey +
            "@ondemand.saucelabs.com:80/wd/hub").build();

        driver.get(baseUrl);
        driver.getTitle().then(function (title) {
            console.log("title is: " + title);
        });
        driver.quit();
        done();
    });
});
```

Instant Test with Best Practices and Sauce Labs Reporting

This example script illustrates the use of several best practices in test design, and will report Pass/Fail status to the Sauce Labs dashboard.

Click here to view the example script

Clone from: https://github.com/saucelabs-training/demo-js/blob/master/web-tests/instant-sauce-mocha-test4.js

web-tests/instant-sauce-mocha-test4.js (source from Sauce Labs Training JavaScript)

```
var webdriver = require('selenium-webdriver'),
    assert = require('assert'),
    username = process.env.SAUCE_USERNAME,
    accessKey = process.env.SAUCE_ACCESS_KEY,
    /* Change the baseURL to your application URL */
    baseUrl = "https://www.saucedemo.com",
    tags = ["sauceDemo", "demoTest", "module4", "nodeTest"],
    driver;


describe('Instant Sauce Test Module 4', function() {
    this.timeout(40000);

    beforeEach(function (done) {
        var testName = this.currentTest.title;
        driver = new webdriver.Builder().withCapabilities({
            'browserName': 'chrome',
            'platform': 'Windows 10',
            'version': '59.0',
            'username': username,
            'accessKey': accessKey,
            'build': 'Onboarding Sample App - NodeJS',
            'name': '4-best-practices',
            /* As a best practice, set important test metadata and
execution options
            such as build info, tags for reporting, and timeout
durations.
             */
            'maxDuration': 3600,
            'idleTimeout': 1000,
            'tags': tags,
        }).usingServer("http://" + username + ":" + accessKey +
            "@ondemand.saucelabs.com:80/wd/hub").build();

        driver.getSession().then(function (sessionid) {
            driver.sessionID = sessionid.id_;
        });
        done();
    });

    afterEach(function (done) {
        driver.executeScript("sauce:job-result=" + (true ? "passed" :
"failed"));
        driver.quit();
        done();
    });

    it('should-open-chrome', function (done) {
        driver.get(baseUrl);
        driver.getTitle().then(function (title) {
            console.log("title is: " + title);
            assert(true);
            done();
        });
    });
});
```

Running Local Tests

Developing websites/apps within your local network is secure and efficient. The drawback to this method is that local assets are not publicly-accessible on the Internet, so the browsers/devices in the Sauce Labs cloud can't load and test your app. The solution is to use Sauce Connect. S auce Connect is a proxy server that creates a secure tunnel connection between the Sauce Labs virtual machine that runs your test and your local network. You can also use Sauce Connect to test applications and websites that are located within your corporate firewall. Sauce Connect is **not** required to run tests on Sauce Labs, only in situations where the application or website you want to test is not publicly accessible.

Running Tests in Parallel

Now that you’re running tests on Sauce, you may wonder how you can run your tests faster. One way to increase speed is by running tests in parallel across multiple virtual machines.

You can run your tests in parallel at two levels, and you can run your tests in parallel across multiple browsers. For example, if you have 10 tests and want to run on five browsers, this would be parallelism of five. You can also run tests across browsers and each test in parallel. Using our previous example this would be more like 50 parallel tests. Doing this requires that your tests are written in a way that they do not collide with one another, as described in our Best Practice topics avoiding external test dependencies and avoiding dependencies between tests.

Check out Using Frameworks to Run Tests in Parallel for more information, and examples of framework setups for Java, Python, and other programming languages.

For more information, check out the example scripts in our GitHub repo.